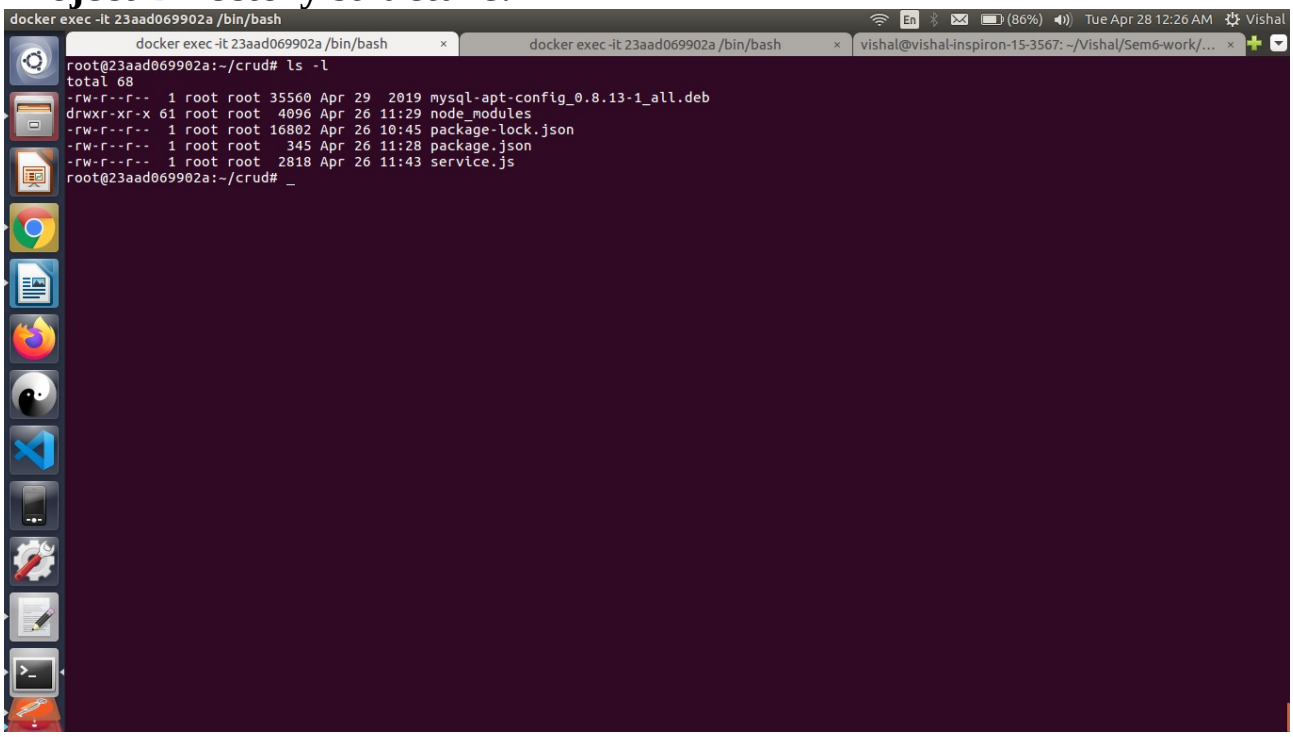
Roll Number:17134
Vishal Kokane
Assignment 002

This assignment is of building REST api using CRUD operations.
I've used nodejs +mysql in base docker image of ubuntu.

Git repository url:https://github.com/vishalkokane264/pucsd2020-devops

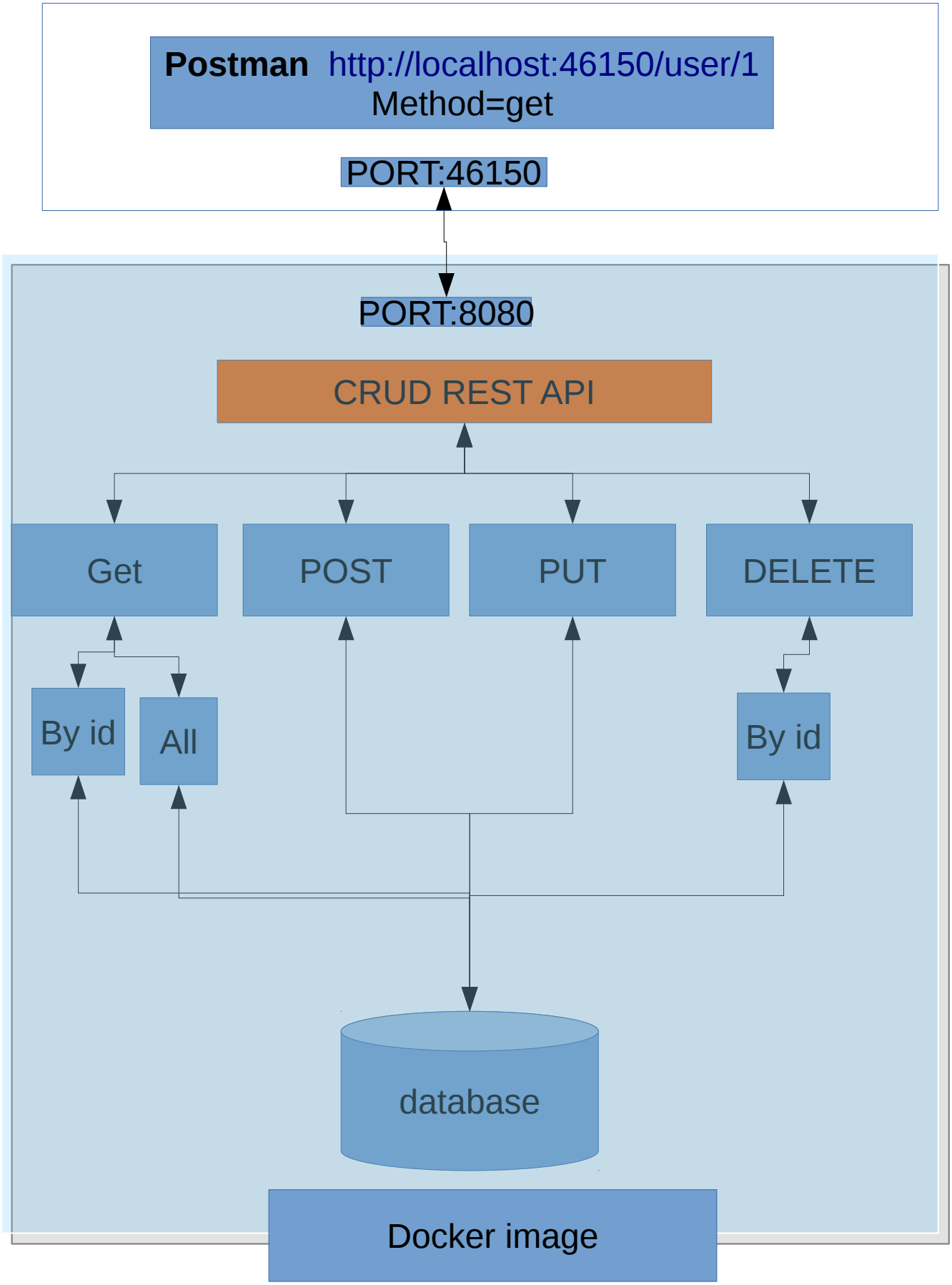
Project Directory structure:



Api Documentation:

Method	URL	Description	
GET by id	http://localhost:46150/user/:id	Get user by user id	
Get (all)	http://localhostL46150/user	Get all users	
POST	http://localhost:46150/user	Add user in database	
PUT	http://localhost:46150/user	Update username using userid	
DELETE(id)	http://localhost:46150/user/id	Delete user using userid	

Flow diagram



dockerfile:

```
FROM crud2:latest

RUN cd
RUN cd /root/crud
RUN /etc/init.d/mysql stop

RUN mysqld_safe --skip-grant-tables &

EXPOSE 8080

CMD ["node","index.js"]
```

---

service.js:

```
// Including the file and initilize
const express = require('express');
const app = express();
const bodyParser = require('body-parser');
const mysql = require('mysql');

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({
        extended : true
}));

const mysql_conn = mysql.createConnection({
    host: 'localhost',
    user: 'vishal',
    database: 'userdb'
});
mysql_conn.connect();

app.get('/',function (req,res){
```

```
        return res.send({error:true, message:
"Rest service using CRUD ops."});
});

app.get('/user', function (req, res) {
    mysql_conn.query('SELECT * FROM user',
function (error, results, fields) {
        if (error) throw error;
        res.end(JSON.stringify(results));
    });
});

// Retrieve user with id
app.get('/user/:id', function (req, res) {

    let user_id = req.params.id;

    if (!user_id) {
        return res.status(400).send({ error: true,
message: 'Enter userID' });
    }

    mysql_conn.query('SELECT * FROM tasks where
id=?', user_id, function (error, results, fields)
{
        if (error) throw error;
        return res.end(JSON.stringify(results));
    });

});


// Add a new user
app.post('/user', function (req, res) {
    var userdata=req.body;
//    console.log(userdata);
    if (!req.body.id) {
```

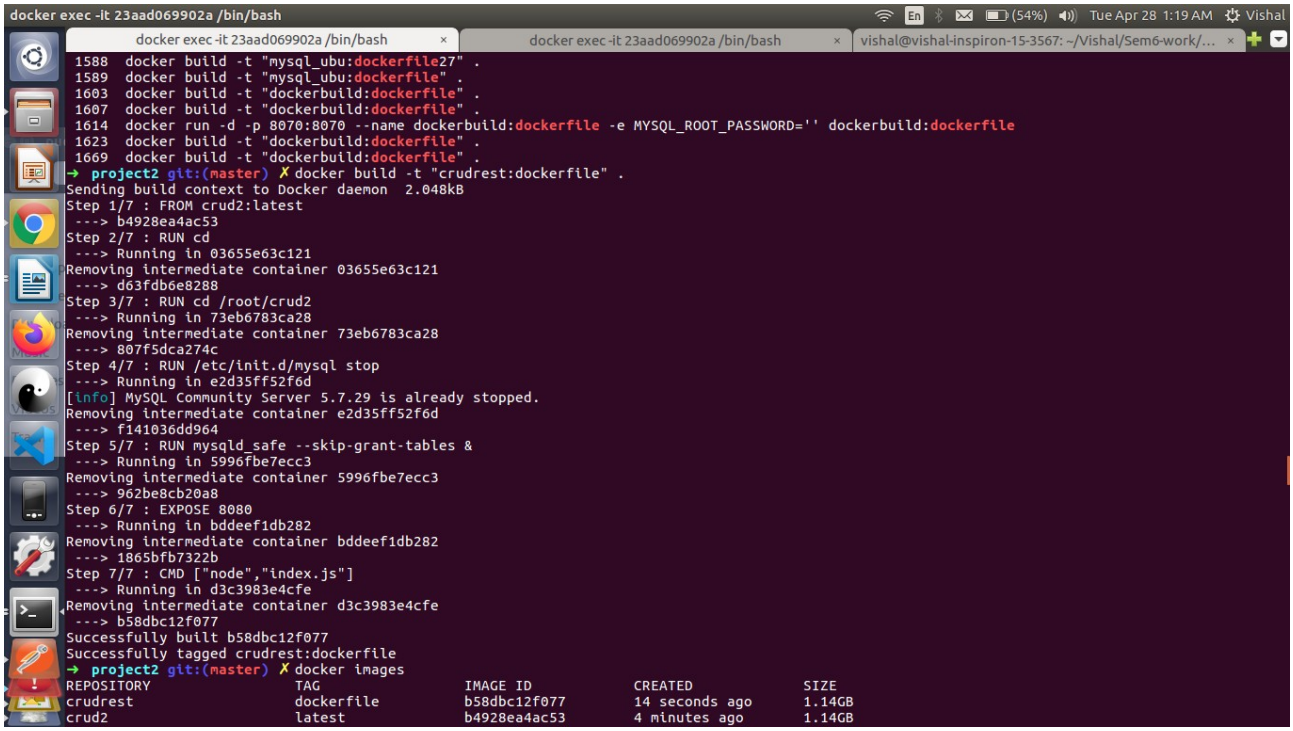
```
        return res.status(400).send({ error:true,
message: 'Please enter userid' });
    }

    mysql_conn.query('INSERT INTO user
SET ?',userdata, function (error, results, fields)
{
        if (error) throw error;
        else{
            console.log("User added
successfully");
            res.end(JSON.stringify(results));
        }
    });
});

//  Update username with id
app.put('/user', function (req, res) {

    let user_id = req.body.id;
    let name = req.body.name;
//    console.log('my user id '+user_id+name);

    if (!user_id || !name) {
        return res.status(400).send({ error: task,
message: 'Enter user id and name' });
    }

    mysql_conn.query("UPDATE user SET name = ?
WHERE id = ?", [name, user_id], function (error,
results, fields) {
        if (error) throw error;
        return res.send({ error: false, data:
results, message: 'User updated successfully.' });
    });
});
```

```
//  Delete user
app.delete('/user/:id', function (req, res) {

    let user_id = req.params.id;

    mysql_conn.query('DELETE FROM user WHERE id
= ?', [user_id], function (error, results, fields)
{
        if (error) throw error;
        return res.send({ error: false, data:
results, message: 'User deleted successfully.' });
    });
});

// All other requests redirect to 404
app.all("*", function (req, res, next) {
    return res.send('Page not found');
    next();
});

//Port must be set to 8080 because incoming http
request are routed from 80 to 8080

app.listen(8080, function(){

        console.log('App is running on port
8080');

});
-----------------------------------------------
```

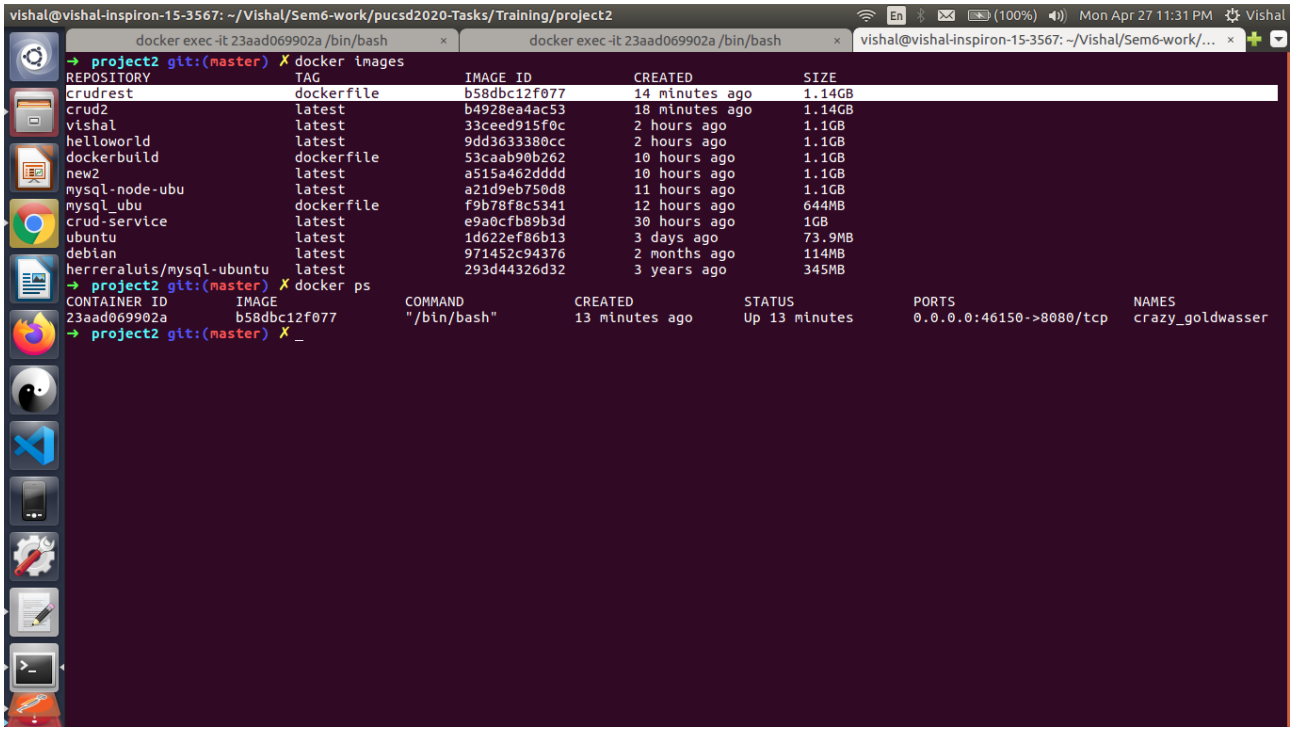
package.json

```
{
  "name": "express-node-rest-project",
  "version": "1.0.0",
  "description": "",
  "main": "service.js",
  "scripts": {
    "test": "echo \"Error: no test
specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.19.0",
    "express": "^4.17.1",
    "mysql": "^2.16.0"
  }
}
```

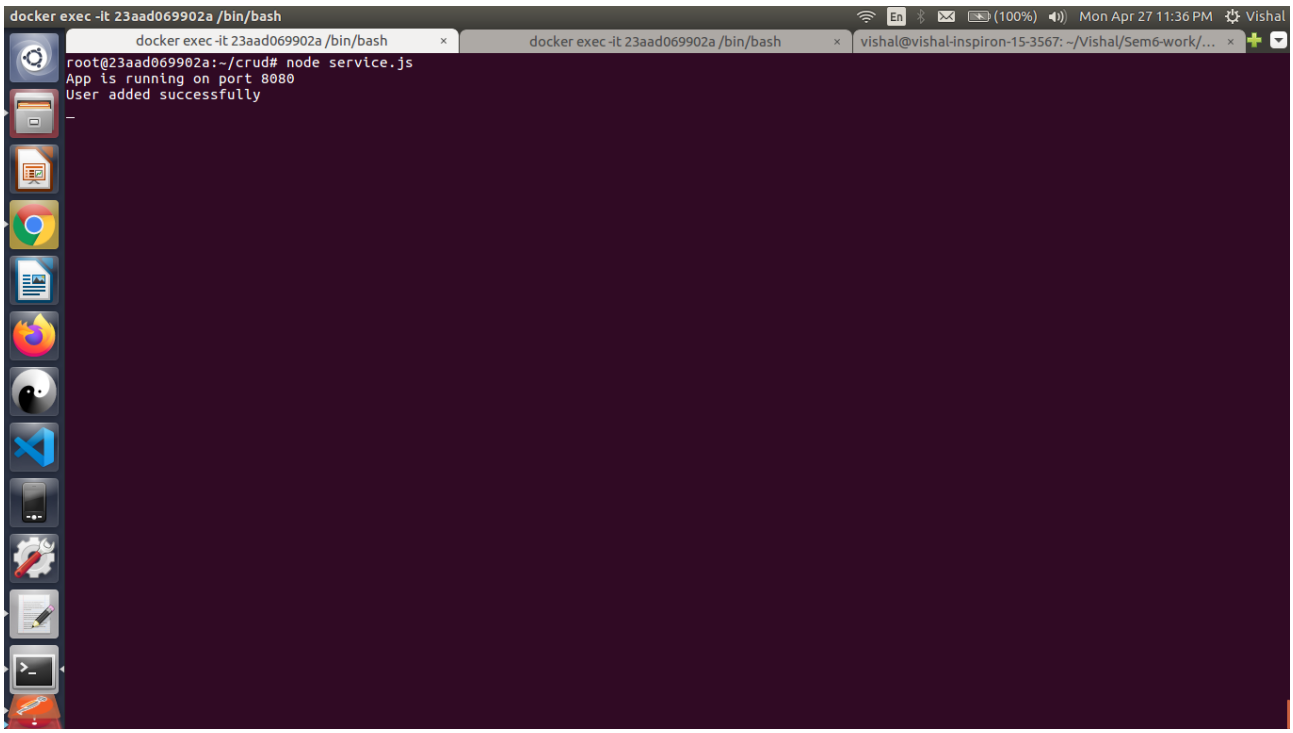
Snapshot:
Running Docker file



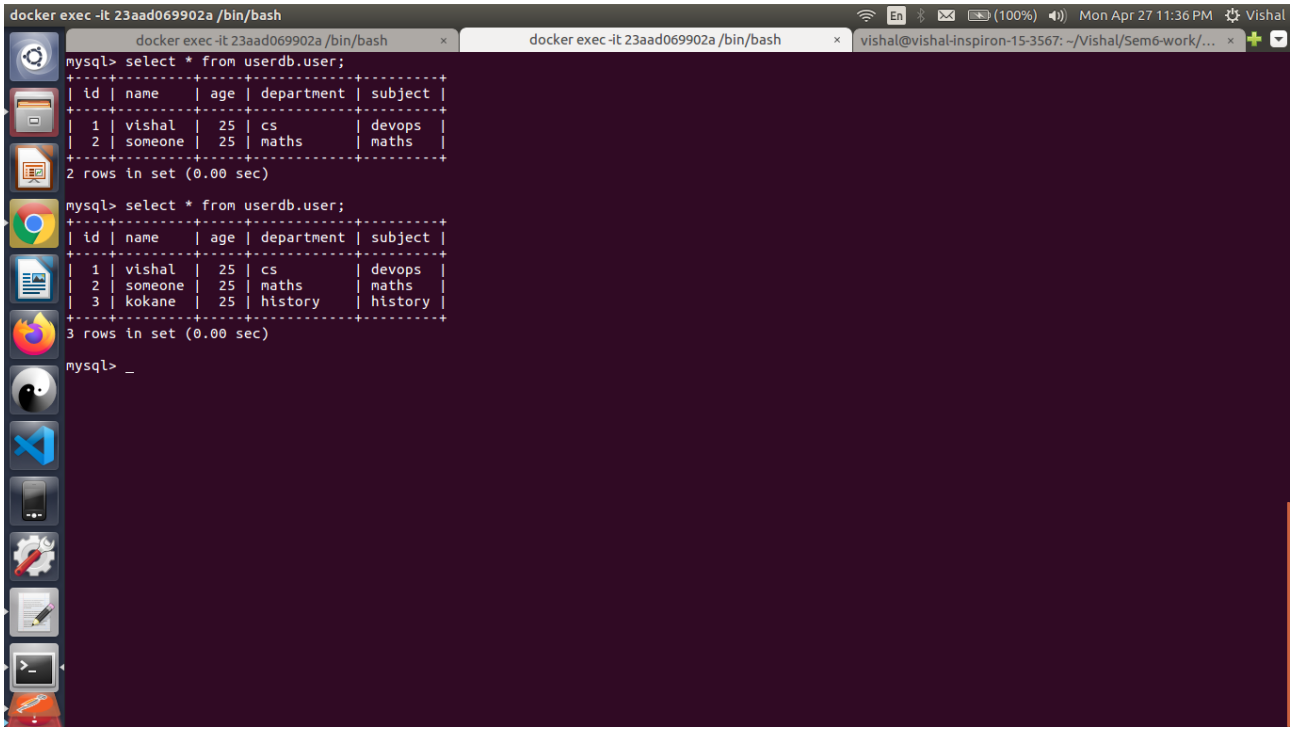
1)Docker images && container running



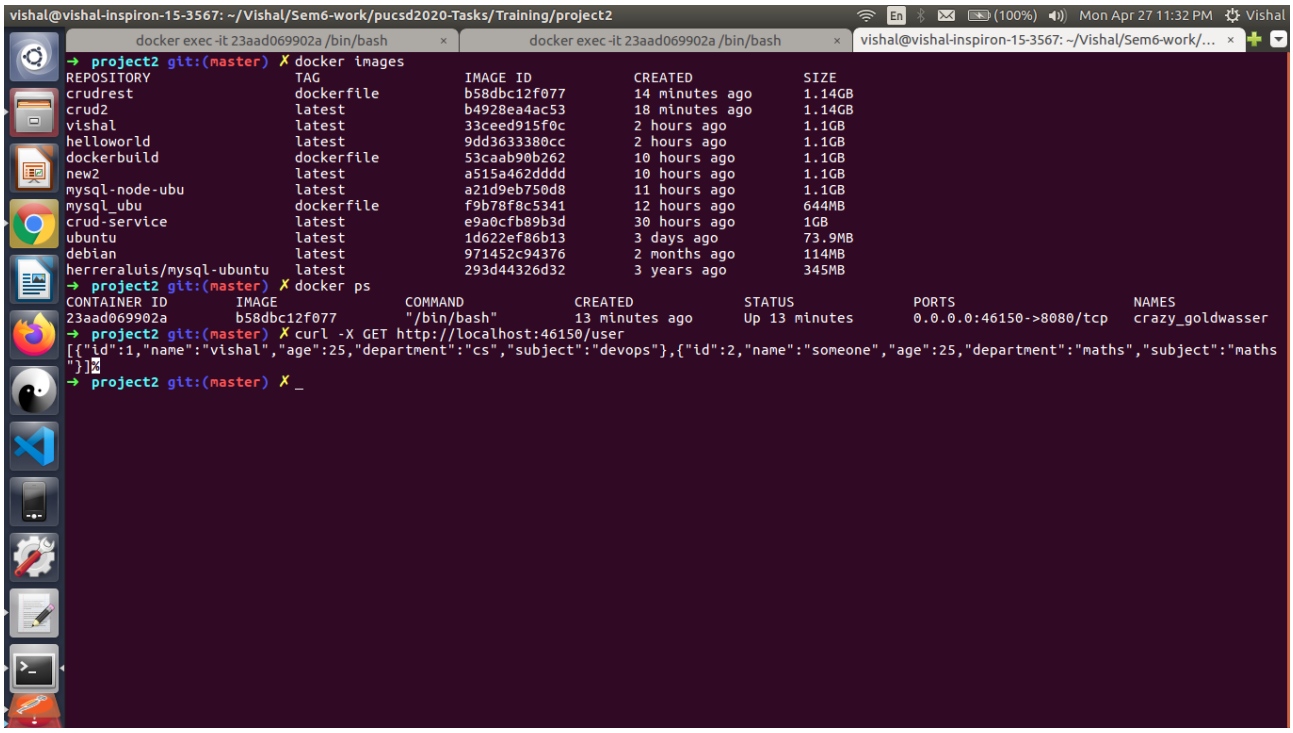
2)Nodejs service running on port 8080



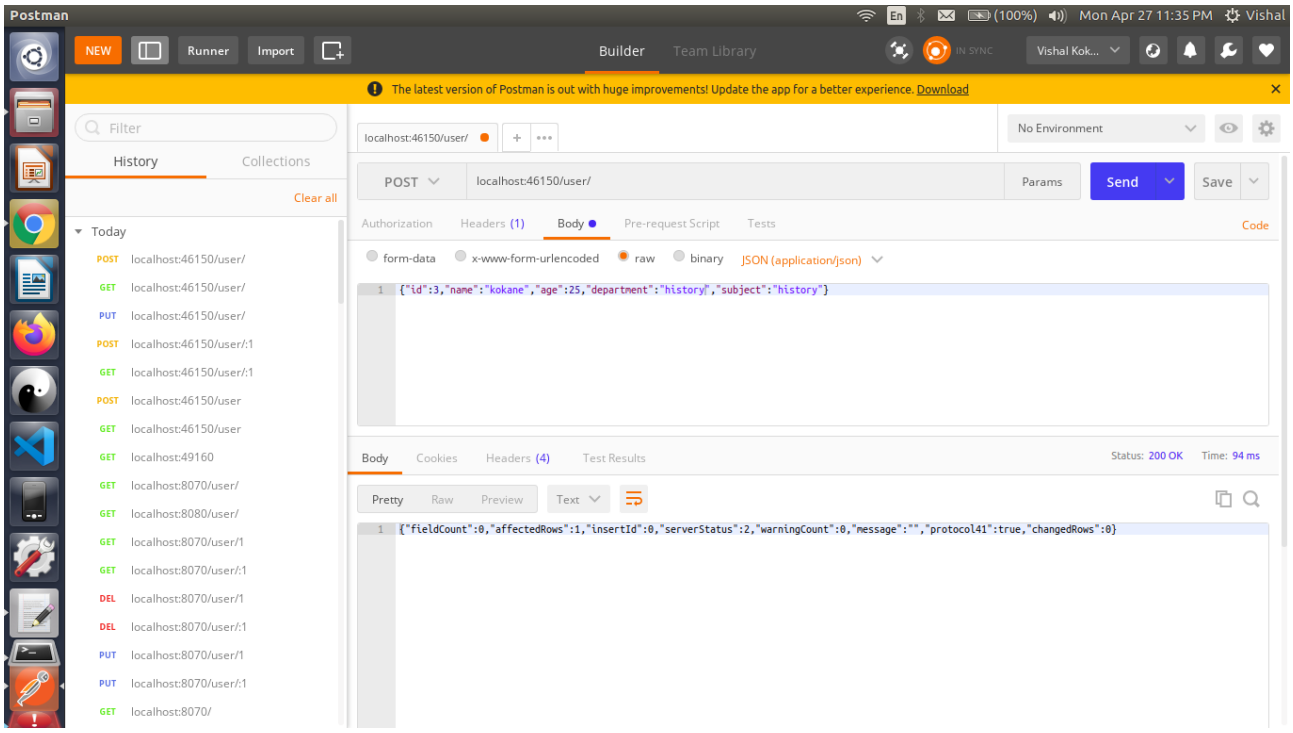
3)Another same container running for checking mysql data



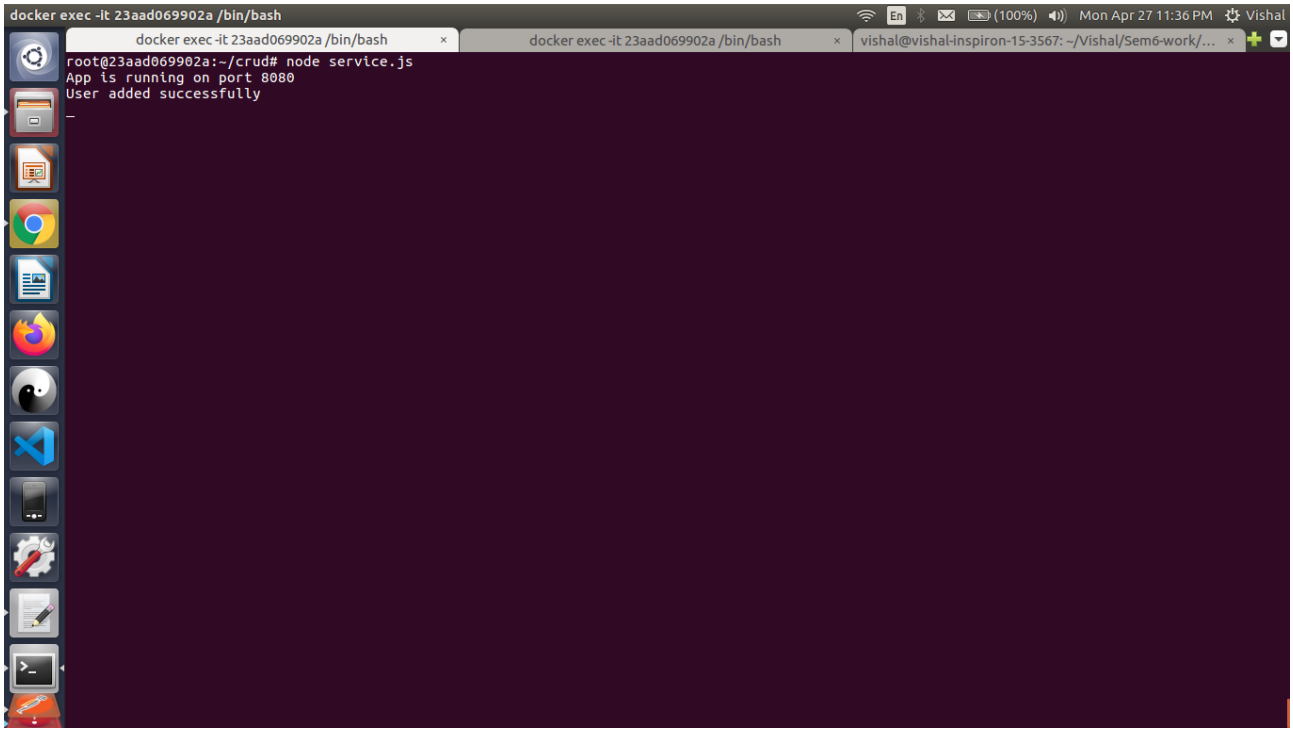
4)Validating service using CURL get request



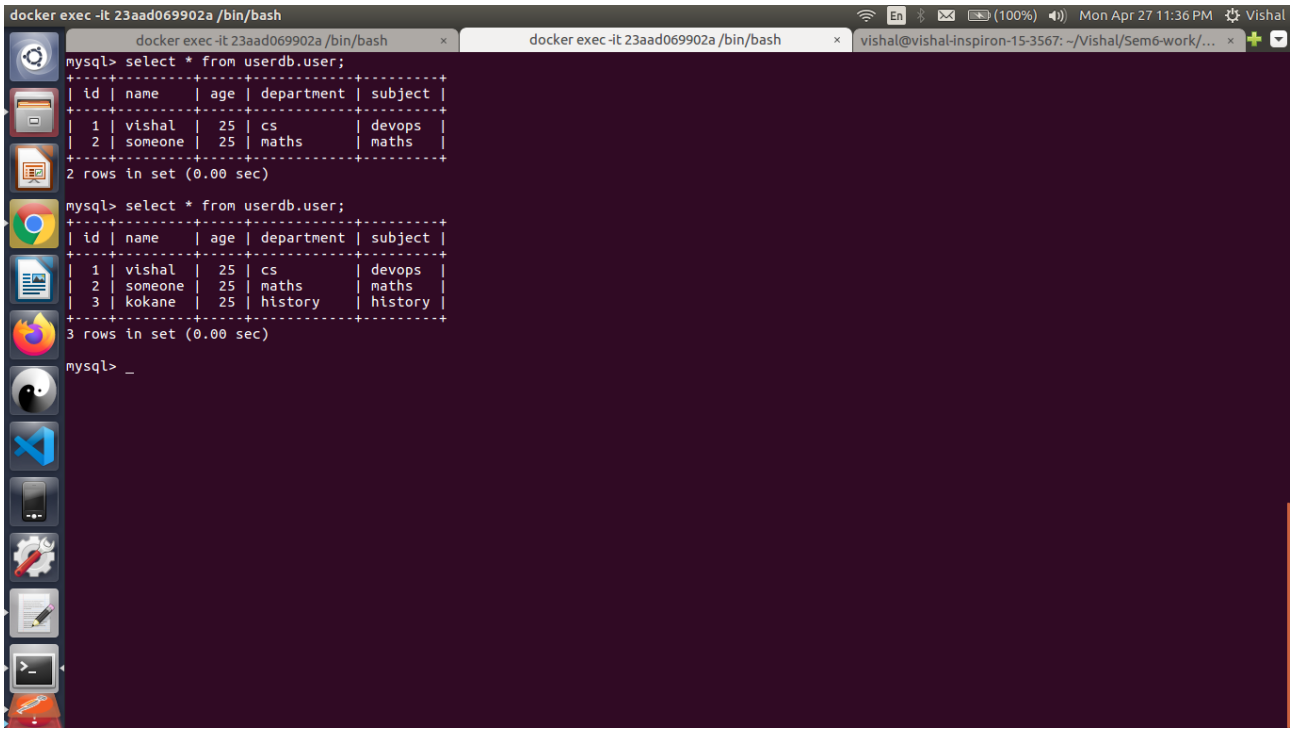
5)Postman POST method:



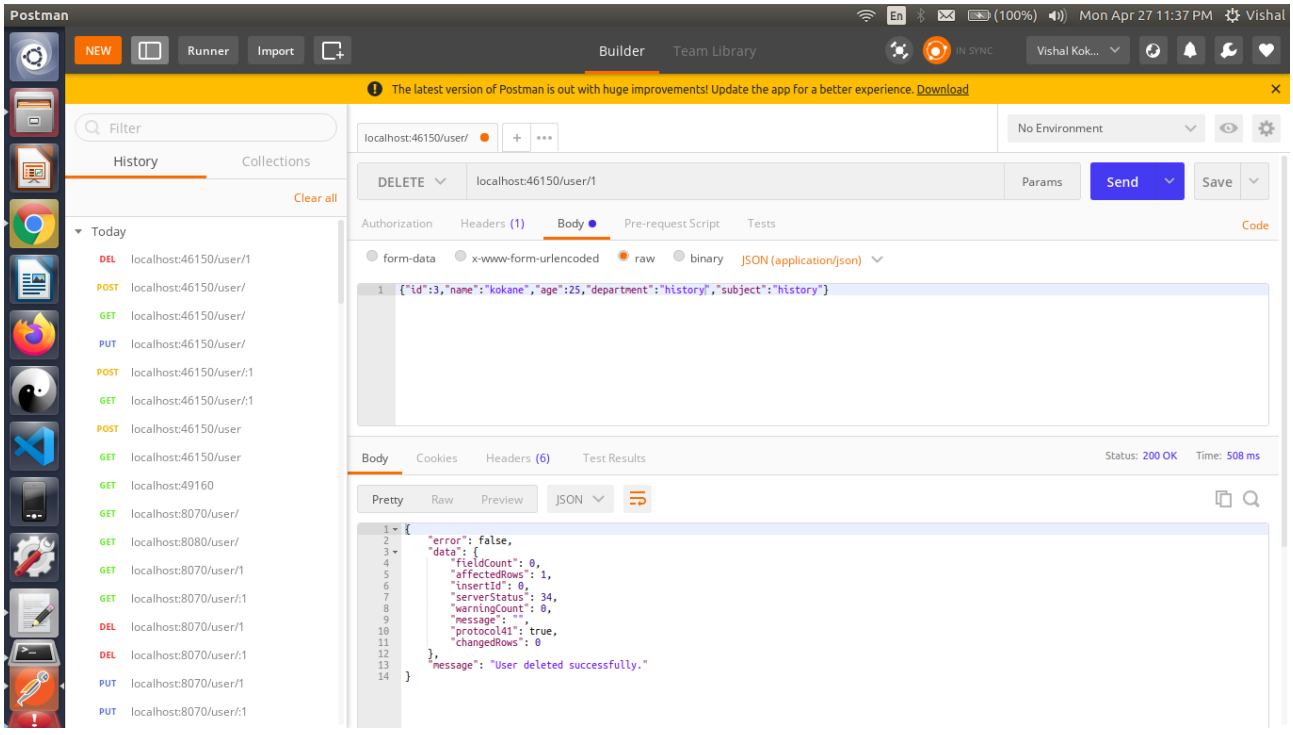
5.1) User added Successfully



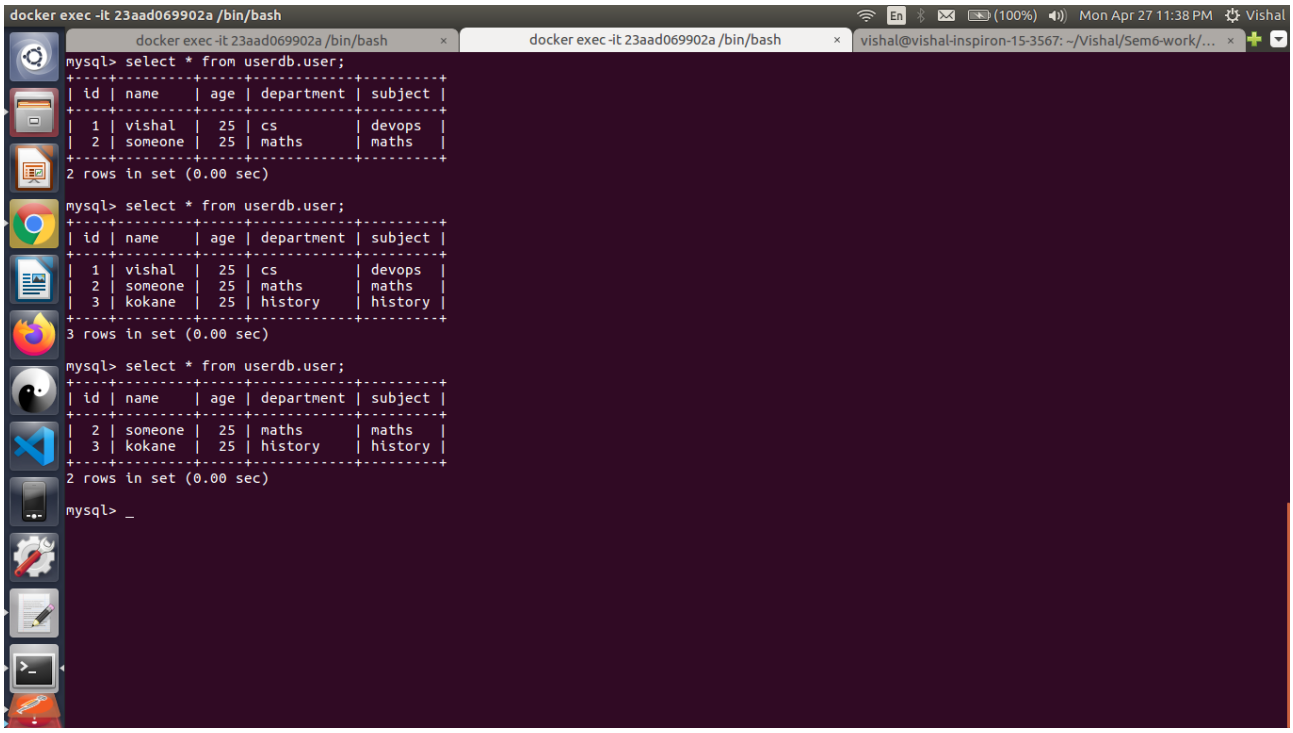
5.2)Mysql result of POST



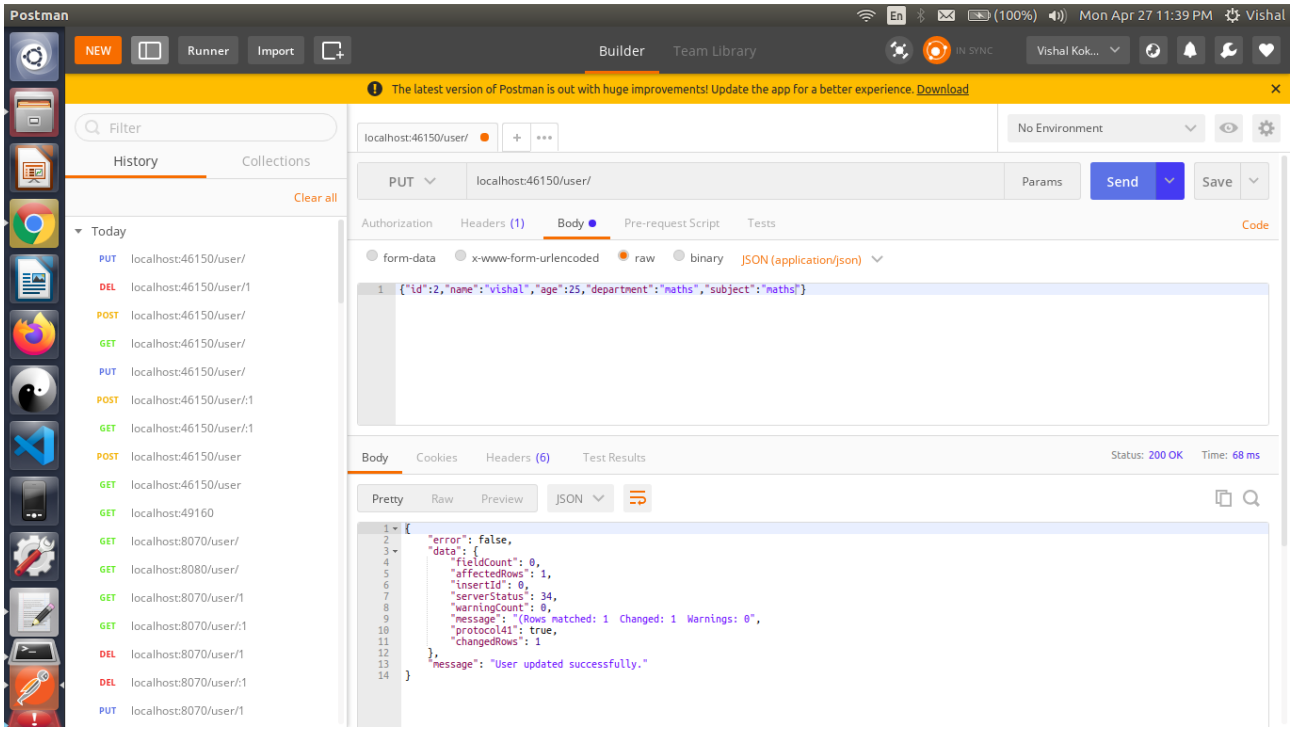
6)Postman delete data with user id =1



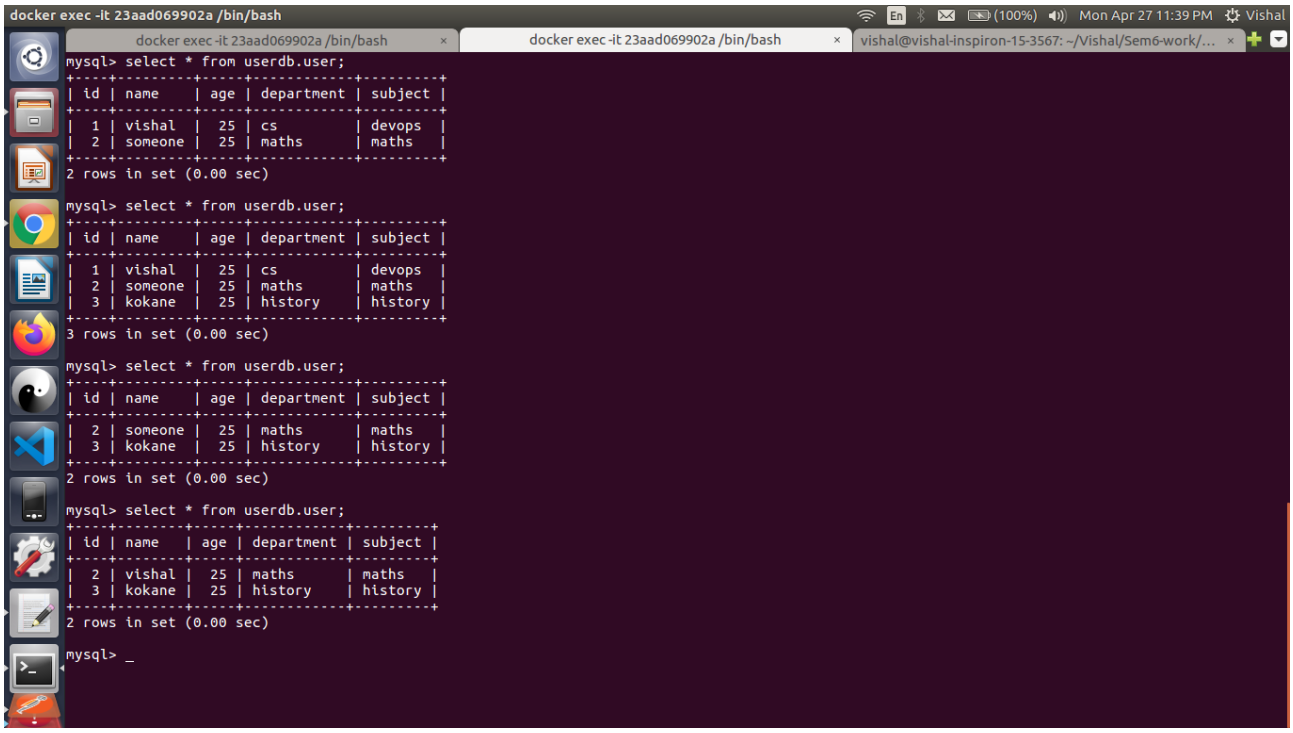
6.2)Mysql status



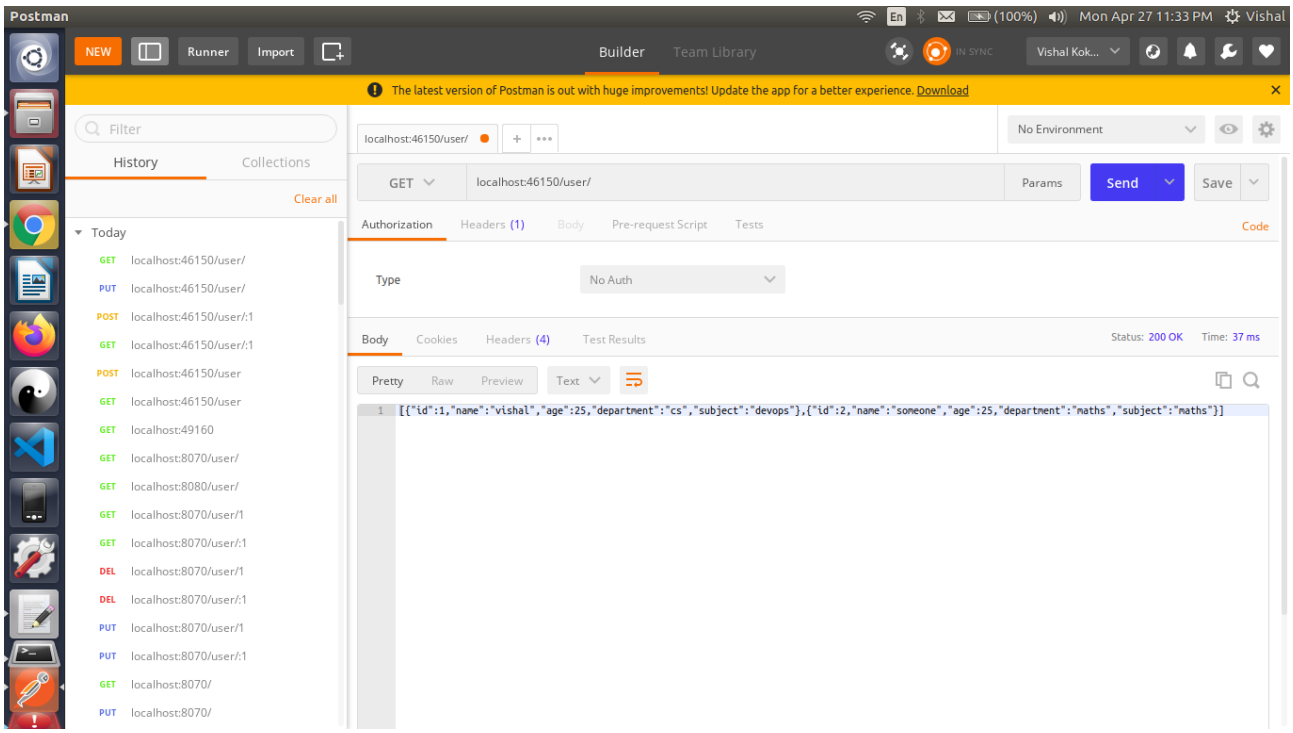
7)Postman PUT :update user name using userid



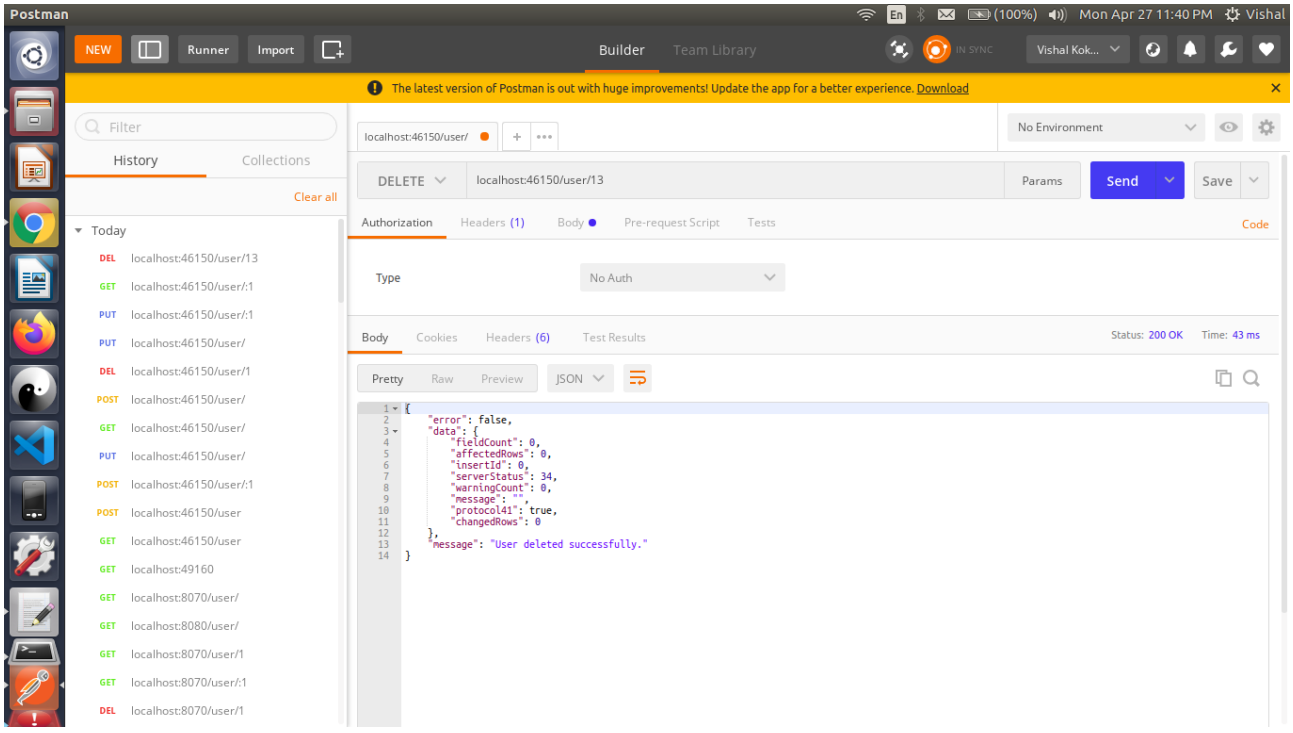
7.2)Mysql update status



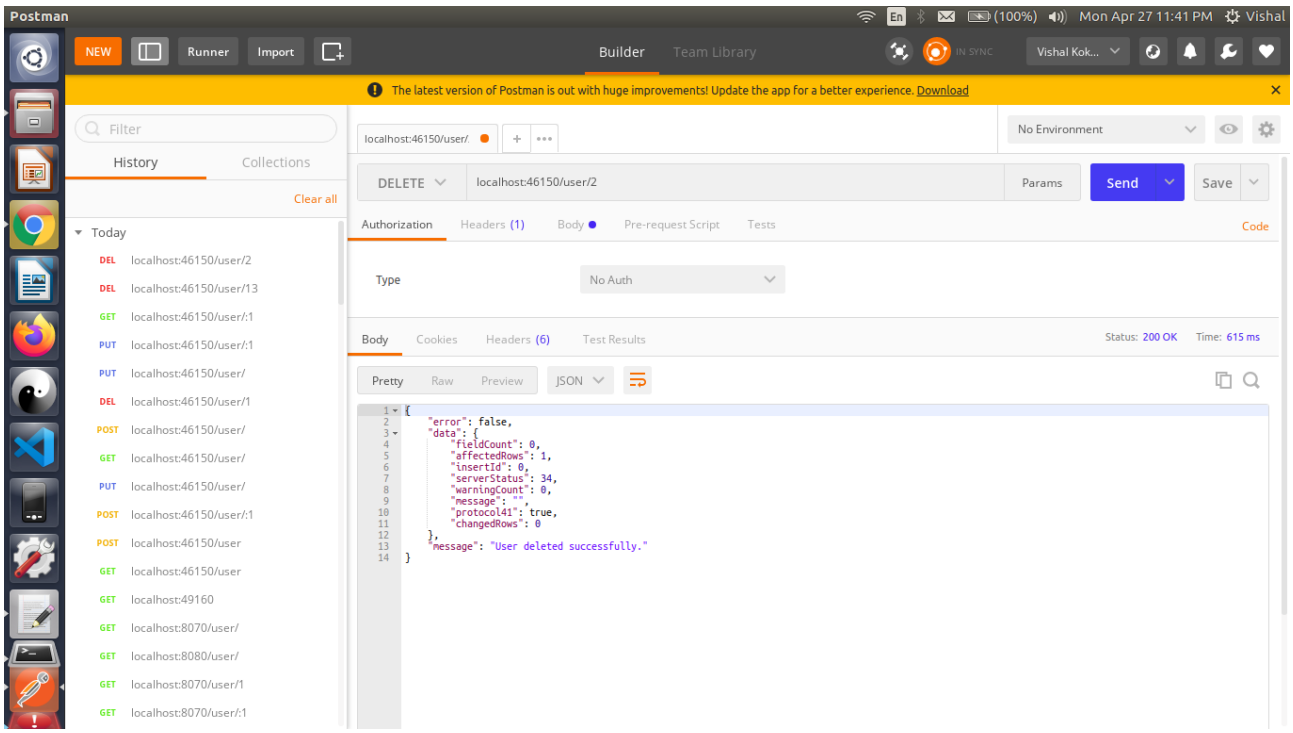
8)Postman get all records



9)Postman delete fail record not exist



10)Postman delete Success


Postman
Mon Apr 27 11:41 PM
Vishal
NEW
Runner
Import
Builder
Team Library
Vishal Kok...
The latest version of Postman is out with huge improvements! Update the app for a better experience. Download
Filter
History
Collections
Clear all
Today
DEL localhost:46150/user/2
DEL localhost:46150/user/13
GET localhost:46150/user/:1
PUT localhost:46150/user/:1
PUT localhost:46150/user/
DEL localhost:46150/user/1
POST localhost:46150/user/
GET localhost:46150/user/
PUT localhost:46150/user/
POST localhost:46150/user/:1
POST localhost:46150/user
GET localhost:46150/user
GET localhost:49160
GET localhost:8070/user/
GET localhost:8080/user/
GET localhost:8070/user/1
GET localhost:8070/user/:1
localhost:46150/user/
No Environment
DELETE
localhost:46150/user/2
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Code
Type
No Auth
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 615 ms
Pretty
Raw
Preview
JSON
{
"error": false,
"data": {
"fieldCount": 0,
"affectedRows": 1,
"insertId": 0,
"serverStatus": 34,
"warningCount": 0,
"message": "",
"protocol41": true,
"changedRows": 0
},
"message": "User deleted successfully."
}

11)POstman Get by id